CRT-2001-01

Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation

CYCLE-BASED NEIGHBOURHOODS FOR FIXED-CHARGE CAPACITATED MULTICOMMODITY NETWORK DESIGN

by

Ilfat Ghamlouche
Teodor Gabriel Crainic
Michel Gendreau

January 2001

PUBLICATION
CRT-2001-01

*CYCLE-BASED NEIGHBOURHOODS FOR FIXED-CHARGE CAPACITATED MULTICOMMODITY NETWORK DESIGN**

by

Ilfat Ghamlouche[1]
Teodor Gabriel Crainic[2]
Michel Gendreau[1]

* This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by the Fonds pour la Formation de Chercheurs et l'Aide à la Recherche (FCAR).

1 Département d'informatique et de recherche opérationnelle et Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 <ilfat, michelg@crt.umontreal.ca>

2 Département de management et technologie, Université du Québec à Montréal, et Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 <theo@crt.umontreal.ca>

Département d'informatique et de recherche opérationnelle - Publication #1188

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2001

Abstract

We propose new cycle-based neighbourhood structures for meta-heuristics aimed at the fixed-charge capacitated multicommodity network design formulation. The neighbourhood defines moves that take explicitly into account the impact on the total design cost of potential modifications to the flow distribution of several commodities simultaneously. Moves are identified through a shortest path-like network optimization procedure and proceed by re-directing flow around cycles and closing and opening design arcs accordingly. These neighbourhoods are evaluated and tested within a simple tabu search algorithm. Experimental results show that the proposed approach is quite powerful and outperforms existing methods reported in the literature.

Keywords : Fixed charge capacitated multicommodity network design, Meta-heuristics, Cycle-based neighbourhoods, Tabu search

Résumé

Nous proposons une nouvelle structure de voisinages pour les méta-heuristiques dédiées au problème de synthèse de réseau multiproduits avec capacités. Les mouvements ainsi définis prennent explicitement en compte l'impact sur le coût du design de modifications simultanées à la distribution de plusieurs produits. L'identification de bons mouvements se fait par l'intermédiaire d'une procédure d'optimisation basée sur une méthode de calcul de chemins les plus courts. Une solution voisine est obtenue par la re-direction de flots dans des cycles et la fermeture/ouverture subséquente d'arcs de design. Nous développons une méthode simple de recherche avec tabous pour tester cette structure de voisinage. Les résultats obtenus indiquent que cette méthode constitue la meilleure heuristique actuellement disponible pour le problème étudié.

Mots-clés : Synthèse de réseau multiproduits avec capacités, Méta-heuristiques, Voisinages par cycles, Méthodes de recherche avec tabous

1 Introduction

The fixed-charge capacitated multicommodity network design formulation (CMND) represents a generic model for a wide range of applications in planning the construction, development, improvement, and operation of transportation, logistics, telecommunication, and production systems (Balakrishnan, Magnanti, and Mirchandani 1997, Magnanti and Wong 1986, Minoux 1986). In these applications, multiple commodities (goods, data, people, etc.) must be routed between different points of origin and destination over a network of limited capacities. Moreover, other than the routing cost proportional to the number of units of each commodity transported over a network link, a fixed cost must be paid the first time the link is used, representing its construction (opening) or improvement costs. The objective of CMND is to identify the optimal design that is, to select the links to include in the final version of the network in order to minimize the total system cost, computed as the sum of the fixed and routing costs, while satisfying the demand for transportation.

Fixed charge network design problems are difficulty (they belong to the NP-hard complexity class, Magnanti and Wong 1986), capacitated ones even more so (Balakrishnan, Magnanti, and Mirchandani 1997) due, among other factors, to the competition of commodities for the limited network capacity and the subsequent multiplication of equivalent-cost flow distributions for a given design, and the difficulty to represent trade-offs between the arc fixed costs and capacities. As a consequence, for now, exact methods cannot solve realistically dimensioned cases (Crainic, Frangioni, and Gendron 2000, Gendron, Crainic, and Frangioni 1998, Holmberg and Yuan 2000). For capacitated problems of any interesting size, only specially tailored heuristics have proved of any help. The simplest of these are drop and add procedures based on reduced cost calculations which determine the marginal value of including or excluding an arc from the network , while the more sophisticated make use of the marginal values of paths in the network. These local search heuristics were applied to particular problem classes, however, and did not attempt to explore past the first local optimum. Recently, Crainic, Gendreau, and Farvolden (2000) proposed a tabu search metaheuristic for the path-based formulation of the problem. The method explores the space of the path-flow variables by using pivot-like moves and column generation. Even though the method produced very impressive results, its search efficiency may be limited by the fact that each move considers the impact of changing the flow of one commodity only (a pivot from one path to another). Moreover, one cannot guarantee that all paths will be considered, neither explicitly, nor implicitly.

The goal of this paper is to propose a new class of neighbourhoods for meta-heuristics aimed at the CMND. The neighbourhood defines moves that may explicitly take into account the impact on the total design cost of potential modifications to the flow distribution of several commodities simultaneously. The fundamental idea is to explore the space of the arc design variables by re-directing flow around cycles and closing and opening design arcs accordingly. Thus, compared to path-based neighbourhoods, not only the move evaluations are more comprehensive (since all commodities on a cycle are explicitly considered), but also the range of moves is broader since flow deviations are no longer restricted to paths linking origins

and destinations of actual commodities. To illustrate the quality of this neighbourhood, we implement two instances in a tabu-based local search meta-heuristic. Computational experiments on problems of various sizes (up to 700 arcs and 400 commodities) show that the tabu search algorithm based on these neighbourhood structures is quite powerful, outperforming already existing methods in solution quality for similar computational efforts.

The contribution of this paper is twofold. First, it introduces a new class of cycle-based neighbourhood structures for meta-heuristics aimed at the CMND, together with efficient procedures to identify good moves. Second, to demonstrate the quality of the cycle-based neighbourhoods, it proposes a very simple tabu search procedure that offers the current best heuristic solutions for the CMND.

The paper is organized as follows. In Section 2, we recall the formulation of the capacitated multicommodity network design problem. Section 3 introduces the concept of cycle-based neighbourhood structures and describes procedures to identify associated "best" moves. The tabu search algorithm is described in Section 4, while calibration and computational results are reported in Section 5. We conclude with perspectives and a number of research avenues.

2 Mathematical Formulation

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network with set of nodes $\mathcal{N}$ and set of directed arcs $\mathcal{A}$. Without loss of generality, we assume that all $(i, j \in \mathcal{A})$ are design arcs. Let $\mathcal{P}$ denote the set of commodities to move using this network, where each commodity p has a single origin $o(p)$, a single destination $s(p)$, and a flow requirement of w^p units between its origin and destination nodes. The arc-based formulation of the CMND can then be written as follows:

$$\min z(x, y) = \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \sum_{p\in\mathcal{P}} \sum_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij}^p \tag{1}$$

$$\text{subject to} \sum_{j\in\mathcal{N}^+(i)} x_{ij}^p - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^p = d_i^p \quad \forall i \in \mathcal{N}, \forall\, p \in \mathcal{P}, \tag{2}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^p \leq u_{ij} y_{ij} \quad \forall (i,j) \in \mathcal{A}, \tag{3}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}, \forall p \in \mathcal{P}, \tag{4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A} \tag{5}$$

where y_{ij}, $(i,j) \in \mathcal{A}$, represent the design variables that equal 1 if arc (i,j) is selected in the final design (and 0 otherwise), x_{ij}^p stand for the flow distribution decision variables indicating the amount of flow of commodity $p \in \mathcal{P}$ on arc (i,j), and

$\mathcal{N}^+(i)$: Set of outward neighbours of node i,
$\mathcal{N}^-(i)$: Set of inward neighbours of node i,
u_{ij} : Capacity of arc (i,j),
f_{ij} : Fixed cost applied on arc (i,j),
c_{ij}^p : Cost of one unit flow of commodity p on arc (i,j),

and

$$d_i^p = \begin{cases} w^p & if \quad i = o(p) \\ -w^p & if \quad i = s(p) \\ 0 & \text{otherwise.} \end{cases}$$

The objective function 1 accounts for the total system cost, the fixed cost of arcs included in a given design plus the cost of routing the product demand, and aims to select the minimum cost design. Constraints 2 represent the network flow conservation relations, while constraints 3 state that for each arc, the total flow of all commodities cannot exceed its capacity if the arc is opened ($y_{ij} = 1$) and must be 0 if the arc is closed ($y_{ij} = 0$). Relations 5 and 4 are the usual non-negativity and integrality constraints for decision variables.

Finally, recall that for a given design vector $\bar{y}$, the arc-based formulation of the CMND becomes a capacitated multicommodity minimum cost flow problem (CMCF):

$$\min z(x(\bar{y})) = \sum_{p \in \mathcal{P}} \sum_{(i,j) \in \mathcal{A}(\bar{y})} c_{ij}^p x_{ij}^p \tag{6}$$

subject to (2) plus

$$\begin{aligned} \sum_{p \in \mathcal{P}} x_{ij}^p &\leq u_{ij} \bar{y}_{ij} \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \\ x_{ij}^p &\geq 0 \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \forall p \in \mathcal{P}. \end{aligned}$$

where $\mathcal{A}(\bar{y})$ stands for the set of arcs corresponding to the design $\bar{y}$.

3 Cycle-based Neighbourhoods

Consider the solution space of the design variables y. A solution to the CMND is then an assignment $\bar{y}$ of 0 or 1 to each design variable, plus the optimal flow of the corresponding multicommodity minimum cost flow problem $x^*(\bar{y})$. Similarly, the objective function value associated to a solution $(\bar{y}, x^*(\bar{y}))$ is the sum of the fixed cost of the open arcs in $\bar{y}$ and the objective function value of the CMCF associated to $x^*(\bar{y})$:

$$z(\bar{y}, x^*(\bar{y})) = \sum_{(i,j) \in \mathcal{A}(\bar{y})} f_{ij} \bar{y}_{ij} + z(x^*(\bar{y})) \tag{7}$$

The classical neigbourhood used to explore such a space changes the value of one, sometimes two, variables at a time (Glover and Laguna 1997). This corresponds to adding an arc to the design, dropping an arc from the network, or swapping the values of an open and a closed arc. Such moves have performed well in related cases such as location problems (e.g., Crainic *et al.* 1993). They fail in the case of capacitated network design problems. The main reason is that, contrary to location problems where, in most cases, paths between origins and destinations are very short (one link, usually) and each site has a significant impact on the design and corresponding flow distribution, in design problems there are many arcs and origin-to-destination paths are many arcs long. An arc is then only one of many and its impact is generally limited. One may often reroute traffic and obtain an almost equivalent solution. Or, as illustrated by arc (a, b) in Figure 1, it may not even be connected to the other currently open arcs, so introducing such an arc into the network has no influence whatsoever on the current solution. One needs, therefore, moves that acknowledge that commodities move on paths and that open and close several arcs simultaneously. The goal of this paper is to introduce such a neighbouring structure. But first, a few basic graph definitions:

1. A simple directed *path* from node i_0 to i_n is a sequence of closed or opened arcs $\{(i_0, i_1), (i_1, i_2), ..., (i_{n-1}, i_n\}$ such that the origin node of each arc is the destination node of the preceding arc in the sequence, and $i_0, ...i_n$ are all distinct nodes. In Figure 1, $\{(h, f), (f, a), (a, b), (b, c), (c, s)\}$ is a path.
2. A simple *chain* is similar to a path except that arcs are not restricted to follow the same direction. In Figure 1, $\{(a, b), (b, c), (c, d)\}$ represents a chain.
3. A (simple) *cycle* is a closed chain, such as $\{(a, b), (b, c), (c, d), (d, e), (e, f), (f, a)\}$ in Figure 1.

3.1 Neighbourhood definition and exploration

The fundamental idea at the origin of the new neighbourhood class is that one may move from one solution to another by

1. Identifying two points in the network together with two paths connecting these points, thus closing a cycle;
2. Deviating the total flow from one path to another such that at least one currently open arc becomes empty;
3. Closing all previously open arcs in the cycle that are empty following the flow deviation and, symmetrically, opening all previously closed arcs that now have flow.

To illustrate, paths $\{(f, e), (e, d), (d, c)\}$ and $\{(f, a), (a, b), (b, c)\}$ in the partial network displayed in Figure 1 form a cycle and flow from the former may be deviated to the latter.

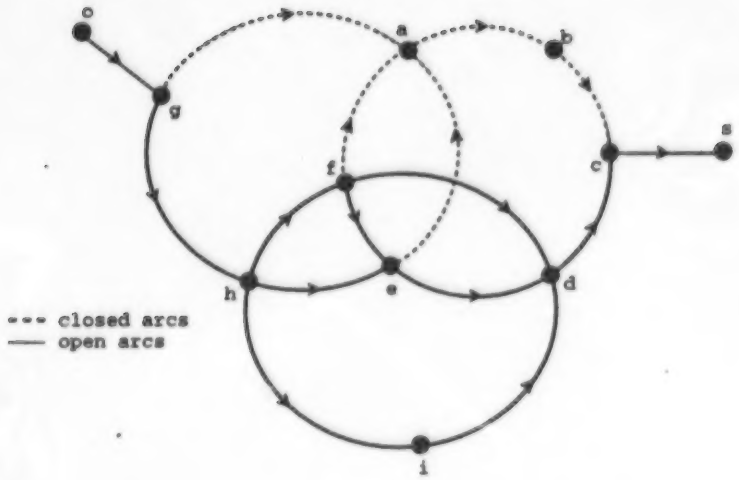


Figure 1: Example of Network, Paths, and Cycles

Three arcs will thus be open, (f,a), (a,b), and (b,c) and, provided sufficient flow may be deviated, at least one previously open arc will be closed. The general neighbourhood structure we propose for the CMND may then be written as

$\mathcal{V}(\bar{y}) = \{$ y: obtained from $\bar{y}$ by complementing the status of a number of arcs following the deviation of flow in a given cycle in $\mathcal{A}(\bar{y})$ $\}$

Such neighbourhoods are huge and their explicit and exhaustive exploration is not practical in most situations. Moreover, the complete evaluation of any design modification involves the resolution of a capacitated multicommodity network flow problem, which rapidly becomes extremely computation intensive. Thus, in order to select the best move out of a given solution, one cannot simply identify explicitly all neighbours, evaluate them, and retain the best one. A more efficient procedure must be implemented that 1) avoids the complete evaluation of every examined move and 2) generates a limited number of cycles that include the "good" moves. One such procedure is presented in the following and it is implemented and tested in Section 4.

Note that not all cycles are of equal interest. We seek, in particular, moves that modify the status of several arcs and that lead to a significant modification of the flow distribution. Therefore, moves that close at least one arc and open new paths for a group of commodities appear attractive.

To close an arc (i,j), one must be able to deviate all its flow. Let the *residual capacity* of a cycle denote the maximum flow one can deviate around the cycle. The residual capacity of any cycle that includes (i,j) must then be at least equal to $\sum_{p\in\mathcal{P}} x_{ij}^p$, the total flow on arc (i,j). Consequently, the cycles of interest to us are those that display a residual capacity equal to one of the values in Γ defined as the set of the total (strictly positive) volumes on the open arcs of the corresponding network:

$$\Gamma(\bar{y}) = \left\{ \sum_{p\in\mathcal{P}} x_{ij}^p > 0 \ : \ (i,j) \in \mathcal{A}(\bar{y}) \right\}. \tag{8}$$

Cycles have thus to be identified on *residual networks* of $\mathcal{A}(\bar{y})$ defined according to $\Gamma(\bar{y})$ and the one leading to the network modification that yields the largest improvement (smallest deterioration, eventually) in equation (7) corresponds to the best move in the neighbourhood of $\bar{y}$. But, again, finding all such cycles and explicitly evaluating the corresponding moves is impractical and some heuristic must be used. We therefore propose a network optimization procedure to implicitly evaluate a large number of cycles according to criteria that approximate the evaluation criterion defined by equation (7). We thus progressively build a set of good candidate neighbours (cycles) among which the best move is then selected.

Let $\mathcal{C} \subseteq \mathcal{A}$ denote the set of *candidate* links, where each arc $(i,j) \in \mathcal{C}$ will be considered as the starting point of cycles. $\mathcal{C}$ may include all the arcs in $\mathcal{G}$. To reduce the computational burned, however, the cardinality of the set will be reduced by various pruning criteria, as illustrated in Section 4. Let $\mathcal{C}(\gamma) \subseteq \mathcal{C}$ include all arcs $(i,j) \in \mathcal{C}$ that can support a movement of γ units of flow. A closed arc may belong to $\mathcal{C}(\gamma)$ only if its capacity is at least γ. For an open arc, either its flow or its residual capacity must be at least γ units. The heuristic procedure we propose to explore the neighbourhood $\mathcal{V}(\bar{y})$ of a given solution $\bar{y}$, given a set of candidate links $\mathcal{C}(\bar{y})$, may then be synthesized as follows:

- Build set $\Gamma(\bar{y})$;
- For each $\gamma \in \Gamma(\bar{y})$
  - Build the γ-residual network as indicated in Section 3.2;
  - For each arc $(i,j) \in \mathcal{C}(\gamma)$ find the lowest cost cycle by using the network optimization procedure of Section 3.3;
- Select and implement the best move;
- Evaluate new solution.

This procedure is implemented in the tabu search procedure described in Section 4. First, however, we define γ-residual networks and describe the network optimization procedure used to find low-cost cycles in residual networks.

3.2 The γ-residual network

The objective is to build a network such that, given a flow value γ, the network optimization procedure of Section 3.3 may 1) identify low cost cycles that support the deviation of at least γ units of flow and 2) explicitly consider the impact of a potential closure or opening of an arc due to the deviation of flow.

To build the *γ-residual network* corresponding to a flow variation of γ units, replace each arc (i,j) of the original network by at most two arcs $(i,j)^+$ and $(j,i)^-$.

Arc $(i,j)^+$ is included only if an additional amount of γ units of flow may pass on arc (i,j), that is if its residual capacity $u_{ij} - \sum_{p\in\mathcal{P}} x_{ij}^p$ is greater or equal to γ. The cost c_{ij}^+ associated to $(i,j)^+$ approximates the cost of routing γ units of additional flow on arc (i,j). It equals the average commodity routing costs on the arc, plus the fixed cost if it is currently closed:

$$c_{ij}^+ = \frac{\sum_{p\in\mathcal{P}} c_{ij}^p}{|P|}\gamma + f_{ij}(1 - (\lceil \min(1, \sum_{p\in\mathcal{P}} x_{ij}^p) \rceil)).$$

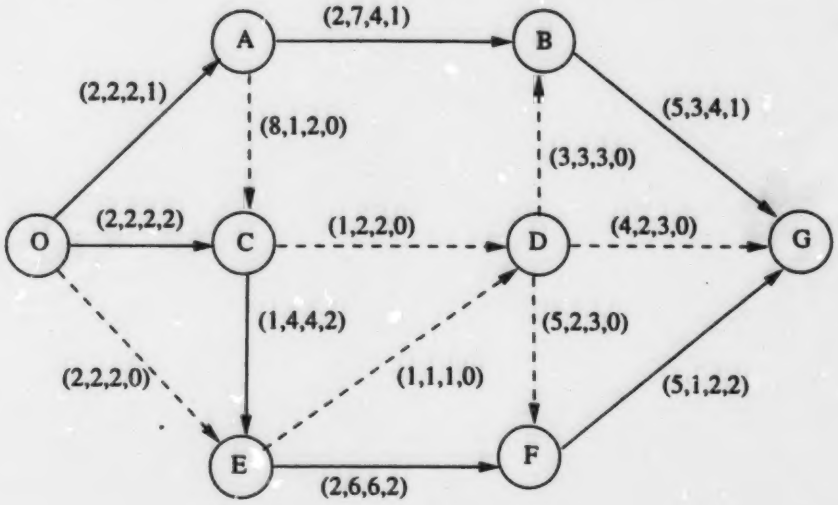
Symmetrically, arc $(j,i)^-$ is not included in the γ-residual network if the total flow on arc (i,j), $\sum_{p\in\mathcal{P}} x_{ij}^p$ is less than γ. Otherwise, we associate to $(j,i)^-$ the cost c_{ji}^- that approximates the value of a reduction of γ units of the total flow (all commodities) currently using arc (i,j). This approximation is computed as the weighted average of the routing costs of the commodities currently using arc (i,j). The fixed cost of (i,j) is then subtracted if the reduction of γ units of flow leaves the arc empty:

$$c_{ji}^- = \begin{cases} -\frac{\sum_{p\in\mathcal{P}} c_{ij}^p x_{ij}^p}{\sum_{p\in\mathcal{P}} x_{ij}^p}\gamma - f_{ij} & \text{if } \sum_{p\in\mathcal{P}} x_{ij}^p = \gamma \\ -\frac{\sum_{p\in\mathcal{P}} c_{ij}^p x_{ij}^p}{\sum_{p\in\mathcal{P}} x_{ij}^p}\gamma & \text{if } \sum_{p\in\mathcal{P}} x_{ij}^p > \gamma \end{cases}$$

To illustrate this procedure, consider the network of Figure 2. In this example, arcs are labeled by the fixed cost, the routing cost (only one commodity is considered), the capacity, and the total flow, respectively. We build a 2-residual network and look for the lowest cost cycle that passes through arc (C,D).

One unit of flow passes on arc (O,A) and its residual capacity is equal to 1. Consequently, one can neither send nor take out 2 units of flow and, thus, there are no arcs between O and A in Figure 3. Arc (O,C) is saturated, so no more flow can be routed from O to C and the associated arc does not appear in Figure 3. On the other hand, reducing the flow of arc (O,C) by 2 units of flow leaves the arc empty, and thus the (C,O) arc is included in the 2-residual network with a cost -6 representing the savings in routing and fixed costs. The rest of the arcs are treated similarly.

Two directed cycles pass through arc (C,D) in the 2-residual network of Figure 3:



(fixed cost, routing cost, capacity, flow)

Figure 2: Network Example

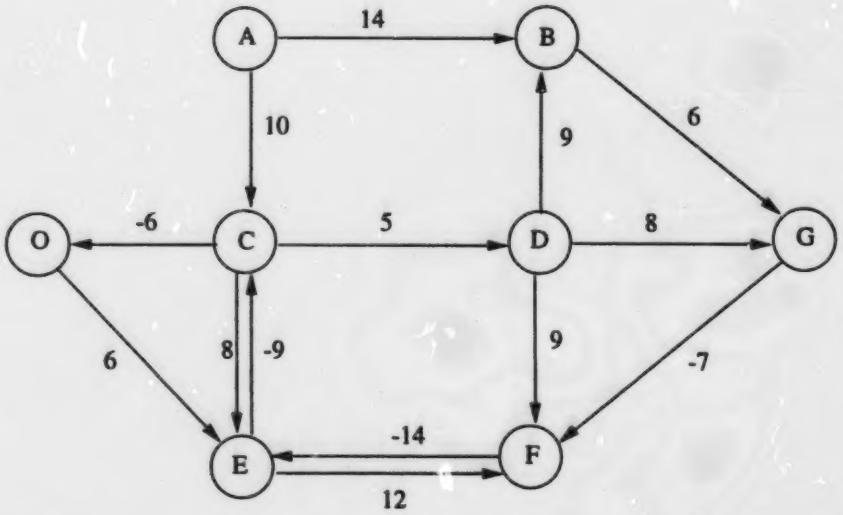


Figure 3: Associated 2-Residual Network

$(D,F),(F,E),(E,C),(C,D)$ and $(D,G),(G,F),(F,E),(E,C),(C,D)$. The latter has the lowest cost with value -17. This cost represents the net change in the objective function of the design formulation when moving 2 units of flow from path $(C,E),(E,F),(F,G)$ to path $(C,D),(D,G)$, opening arcs (C,D) and (D,G), and closing arcs $(C,E),(E,F)$ and (F,G) in the original network. This neighbour is thus reached by a complex move that involves opening two arcs $((C,D),(D,G))$ and closing three others $((C,E),(E,F),(F,G))$.

3.3 Heuristic to identify low cost cycles

The enumeration of all cycles passing through a given arc in a γ-residual network is expensive. Since one is interested in the lowest cost cycle only, one would like to find this particular cycle without enumerating all the others. A network optimization procedure is proposed for this purpose.

The main idea is to find for a given arc (i,j) the shortest path from j to i to complete the cycle. Yet, since the γ-residual network may contain negative cost directed cycles, finding a shortest path is NP-hard (Ahuja, Magnanti, and Orlin 1993). Consequently, we do not attempt to solve the problem exactly. We rather develop a heuristic based on a modification of the shortest path label-correcting algorithm that avoids getting trapped in negative directed cycles and allows to find low-cost cycles.

Label-correcting (Ahuja, Magnanti, and Orlin 1993) is one of the algorithms designed to find shortest paths between a source node and all other nodes in a network which contains no negative directed cycles. The algorithm starts by labelling each node j of the network with $d(j)$, the current distance (cumulated cost) in the shortest path tree from the source to the node. Nodes are also labelled with their respective predecessors in the current shortest paths, $pred(j)$, in order to be able to retrace the shortest paths. The algorithm then proceeds by decreasing, at every iteration, the distance label of a node j in order to satisfy the shortest path optimality conditions

$$d(j) \leq d(i) + c_{ij} \quad \forall (i,j) \in \mathcal{A},$$

where c_{ij} denotes the cost of arc (i,j).

Label-correcting algorithms operate by maintaining a LIST of nodes i that may be the origin of links for which the optimality condition is not yet satisfied. At each iteration, one node i is removed from LIST and the nodes $j \in \mathcal{N}^+(i)$ are scanned. If $d(j) > d(i) + c_{ij}$, $d(j)$ is set to $d(i) + c_{ij}$ and j is added to LIST (if not already included). The algorithm terminates when LIST is empty.

In the presence of negative-cost directed cycles, the algorithm gets trapped and keeps cycling. To avoid this, we modify the label-correcting algorithm to explicitly verify if a scanned node j already belongs to the current shortest path from the source node to its predecessor node i, in which case the label of node j is not modified (we forbid node j to be labeled from node i). The label-correcting-based heuristic then becomes:

$d(s) = 0$, $pred(s) = 0$, $d(j) = \infty$ for each $j \neq s$; LIST $= \{s\}$

While LIST $\neq \phi$

Remove an element i from LIST

For each node j in its forward star $\mathcal{N}^+(i)$

If $(d(j) > d(i) + c_{ij})$ and (j is not in the path from s to i)

- $d(j) = d(i) + c_{ij}$;
- $pred(j) = i$;
- If $j \notin$ LIST then add j to LIST.

The path generated by this heuristic is not necessarily the shortest path. Computational results show, however, that the heuristic produces very good cycles.

4 Tabu Search with Cycle-based Neighbourhoods

Tabu search is an iterative meta-heuristic (Glover 1986, 1989, 1990) that has consistently been successful in addressing hard optimization problems (Glover and Laguna 1997), including the CMND (Crainic, Gendreau, and Farvolden 2000). To evaluate the concepts introduced in the previous section, we developed a simple tabu search-based local search procedure that integrates two versions of the cycle-based neighbourhood: One that considers the flow of all commodities when determining cycles and a second one that refines the search by implementing moves resulting from the deviation of the flow of one commodity only at a time. We first present the basic tabu search framework we use. Then, we examine how the procedure deals with infeasible solutions and how the search is intensified following a move to a "good" solution.

4.1 The tabu search procedure

Following an initialization phase, the tabu search procedure explores the y solution space using a simple local search framework based on the neighbourhood defined in Section 3. At each iteration, the best non-tabu move is determined and implemented regardless whether it improves the overall solution or not. A short-term tabu memory is used to record characteristics of visited solutions to avoid cycling. When a particularly good solution is encountered, the search is intensified using a particular implementation of cycle-based neighbourhoods that consider the flow distribution of one commodity only. A solution is considered particularly good when it either improves the objective value of the best known solution or is within a pre-defined percentage of this value. The method terminates whenever a predefined stopping criterion (number of iterations, CPU time, etc.) is met.

To select the "best" move in the neighbourhood of a given solution $\bar{y}$, the procedure first determines the set of the flow deviation values $\Gamma(\bar{y})$ as defined by (8). To reduce the computational burden, the set of candidate arcs C is restricted to a random subset of closed arcs. Then, following the framework detailed in Section 3.1, a γ-residual network is built for each value $\gamma \in \Gamma(\bar{y})$ (see Section 3.2), and a low-cost cycle is found for each arc $(i,j) \in C(\gamma)$ (see Section 3.3). The lowest overall cycle is then considered as the "best" *local search move* from the current solution. An outline of the tabu search procedure follows.

Initialization

Generate an initial feasible solution and initiate the *BestSolution* and *CurrentSolution* to its objective value.

Main local search loop

While a stopping criterion is not met

- Determine sets C and $\Gamma(\bar{y})$ (as well as sets $C(\gamma)$) for the current solution $\bar{y}$.
- Determine the best overall cycle
- Move to the new solution by opening and closing the appropriate arcs; Determine the solution value of the new solution by solving exactly the associated capacitated multicommodity network flow problem;
- Assign a tabu status to each complemented arc;
- If the solution in infeasible, perform a *restoration* phase;
- If
$$\frac{CurrentSolution - BestSolution}{BestSolution} \leq IntensGap$$
the current solution is a "good" solution and an *intensification* phase is performed.
- If $CurrentSolution < BestSolution$ update *BestSolution*

Short-term tabu memory is used to prevent the search from cycling. Thus, the arcs that are opened or closed receive a tabu status that forbids the reversal of the move for a given number of iterations. The tabu memory is also updated following local search and intensification moves.

An initial feasible solution is produced by opening all arcs and solving the corresponding flow problem. All arcs with flow are then considered open, while all unused arcs are closed. An intensification phase is then performed until negative cycles are no longer detected for any commodity.

4.2 Restoration from Infeasible Moves

The solution produced by a local search move might be infeasible. This follows from the fact that commodities are aggregated when local search moves are determined on γ-residual networks. Consequently, in the new network configuration that follows the closing and opening of arcs in the original network, commodities might follow different paths than the expected ones (i.e., those in the local search cycle-move).

To illustrate, consider the partial network displayed in Figure 4 where three commodities p_1, p_2, and p_3 share the network. Let (x, y, z, t) denote respectively the fixed cost, routing cost, capacity, and flow on each arc and suppose that $(A, D, 2)$, $(E, G, 2)$ and $(B, F, 2)$ represent the origin, destination, and demand of commodity p_1, p_2, and p_3, respectively. The current solution routes 2 units of flow of commodity p_1 on path $(A, B), (B, C), (C, D)$, 2 units of flow of commodity p_2 on path $(E, C), (C, F), (F, G)$, and 2 units of flow of commodity p_3 on path $(B, C), (C, F)$. The objective function value associated to the current solution equals 1021.

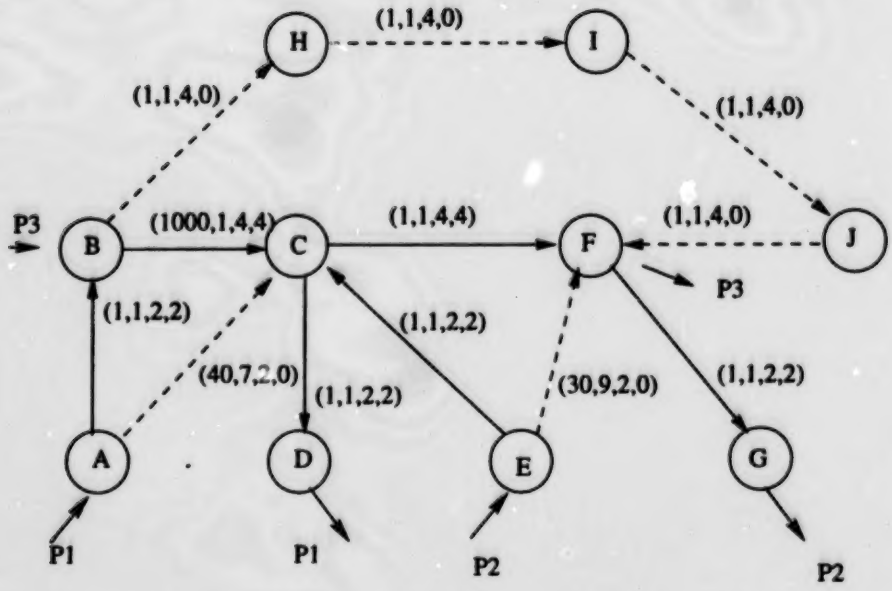
In this example $\Gamma = \{2, 4\}$. Assume all closed arcs are in C and an empty tabu list. Then, the lowest cost cycle in the 4-residual network of Figure 5 shows an improvement of -989. In the 2-residual network (not shown), the lowest cost cycle yields a deterioration of the objective function. Consequently, the procedure would retain the cycle $(B, H), (H, I), (I, J), (J, F), (F, C), (C, B)$ and would move to the best neighbour by opening arcs (B, H), (H, I), (I, J), and (J, F) and closing the arcs without flow (B, C) and (C, F). The resulting network is infeasible since the demands of commodities p_1 and p_2 may no longer be satisfied.

To detect infeasible solutions, artificial arcs between the origin and destination nodes of each commodity may be added to the network. These arcs receive capacities equal to the demand of the associated commodity and high routing costs to ensure that an artificial arc will bear flow only if the solution is infeasible. Then, when the solution produced by a local search move is infeasible, a restoration phase is undertaken in order to reach a solution in the feasible domain. This phase is similar to the neighbourhood local search, except that the set C is made exclusively of artificial arcs with a positive flow, while the set Γ is restricted to the flow of commodities routed on these artificial arcs.

4.3 Intensification

The intensification phase is called each time a local search move yields a "good" solution that is, a solution that improves the best overall solution or is close to it. This phase searches to improve the solution further by iteratively modifying the flow distribution of one commodity only.

The neighbourhood definition of Section 3 is adapted for the intensification phase: Moves



(fixed cost, routing cost, capacity, flow)

Figure 4: Example of Infeasible Move: Original Network

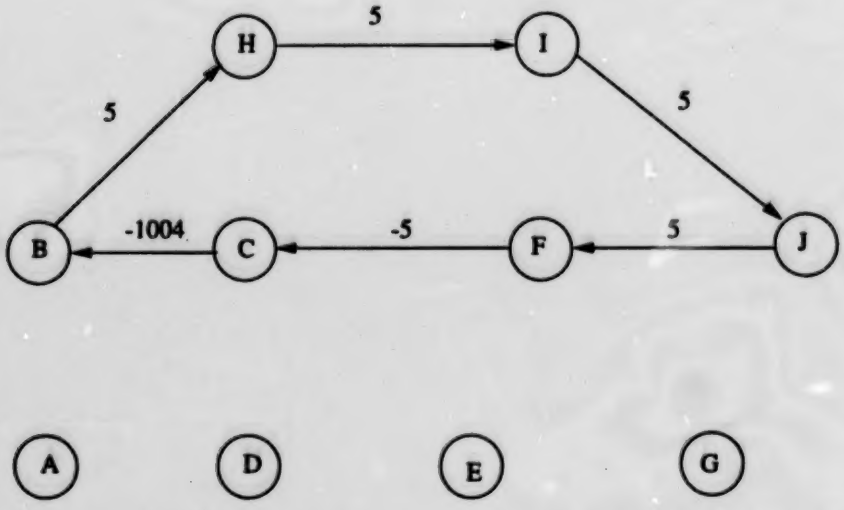


Figure 5: Example of Infeasible Move: Residual Network

are detected by letting the flow of one commodity move around negative directed cycles while the flow of the other commodities is kept fixed. The set Γ is instantiated for each commodity (denoted Γ^p) to contain the flow of the particular commodity that exist on each arc of the network. Moreover, only improving moves are accepted during this phase to ensure that the chosen neighbour will always yield a feasible solution better than the current one. An outline of the intensification phase follows.

Intensification Phase

Repeat until no negative cycle is detected in any γ-residual network
For each commodity $p \in \mathcal{P}$ and flow value $\gamma \in \Gamma^p$

- Build the γ-residual network
- Find a low cost cycle for each arc $(i,j) \in \mathcal{C}(\gamma),\ \gamma \in \Gamma^p$
- Select the best overall cycle;
  Denote γ_{best} the associated volume of deviated flow;
- If the selected cycle is improving (negative cost)
  1. Update the solution:
     - Modify the flow around the selected cycle by a quantity γ_{best};
     - Open/close appropriate arcs;
     - Update the cost of the current solution by adding the cost of the cycle (without solving the CMCF).
  2. Assign a Tabu status to each complemented arc.

Solve exactly the CMCF associated to the current design.

5 Computational Results

Experiments have been performed to evaluate the performance of the simple tabu search meta-heuristic and, more generally, of the cycle-based neighbourhood concept we propose. To ensure meaningful comparisons, we experiment on the same two sets of problem instances also used by Crainic, Gendreau, and Farvolden (2000).

The computer code is written in C++. The exact evaluation of the capacitated multi-commodity network flow problems is done using the linear programming solver of CPLEX 6.5 (1999REF). Unless indicated otherwise, tests were conducted on a Sun Ultra-60/2300 workstation with 2 Gigabyte of RAM, operating under Solaris 2.6. Computing times are reported in seconds.

5.1 Calibration

A calibration phase was conducted to determine appropriate values of key parameters of the tabu search meta-heuristic:

- The size of the neighbourhood set explored at each iteration. This size depends on the dimension of the candidate set C that is, on the percentage of closed arcs randomly selected. Two values, 50% and 70% of the closed arcs, have been tested.
- The length of the tabu tenure of arcs opened and closed following a local search or an intensification move. Four values - 1, 2, 3, and 5 - were considered initially. The 1 and 5 values were rapidly dropped, however, since cycling was observed for a tabu tenure length of 1, while for a value equal to 5 the quality of the solutions started to decrease.
- The threshold used to determine a "good" solution: *IntensGap* equal to 7%, 9%, and 11% of relative improvement.

Calibration experiments were conducted the same problem instances used by Crainic, Gendreau, and Farvolden (2000) to calibrate their path-based tabu search meta-heuristic. The 10 problems cover network sizes from 100 to 700 design arcs and from 10 to 400 commodities. They also display relatively high fixed costs compared to routing costs and tightly capacitated. Moreover, since a random number generator controls the selection of arcs in C, each run was repeated 3 times. The impact of the random seed used was not major, but large enough to make it unsafe to draw conclusions from a single run.

Each parameter combination was ranked for each problem instance according to the average gap (over 3 runs) relative to the best known solution (that of the branch-and-bound procedure of CPLEX 6.5, when available, or that obtained by Crainic, Gendreau, and Farvolden 2000, otherwise). A score of 10, 9, 8, .. , 2, 1 is assigned to each of the first ten places, respectively. The performance of each parameter setting is then aggregated over all runs: scores are summed up, while average gaps and CPU solution times are averaged. Table 1 displays for each parameter combination these aggregated results (over the set of 360 runs: 10 problems tested 3 times for 12 parameter settings). The first column holds the parameter setting, the second and third columns present the global average gaps and CPU times, respectively, while the last column displays the total score.

The results in Table 1 display two parameter settings that achieve the same highest aggregated score. The final choice was then made on the other performance measures and the parameter combination C=50%, tabu tenure= 2, *IntensGap*= 9% was selected since it offers the lowest global average gap and CPU time. Two additional comments are prompted by the results displayed in Table 1. First, that global averages may be hide important variations and may be misleading. Thus, the lowest average gap appears for the fourth ranking parameter setting (70%, 2 and 7%). Secondly, one notices that more than one

Table 1: Parameter Setting Performances

Parameter Setting	Average Gap	Average CPU	Score
50%, 2, 0.09	1.93%	10814.34	68
70%, 2, 0.11	2.07%	13068.72	68
70%, 2, 0.09	2.09%	13011.97	63
70%, 2, 0.07	1.84%	13168.14	62
70%, 3, 0.07	1.96%	12427.75	59
50%, 3, 0.07	2.27%	10984.97	46
50%, 2, 0.07	2.30%	11480.44	45
70%, 3, 0.09	2.62%	12826.49	40
70%, 3, 0.11	2.95%	13528.20	35
50%, 3, 0.11	2.83%	11530.40	26
50%, 2, 0.11	2.72%	10929.73	24
50%, 3, 0.09	2.81%	11172.91	14

parameter setting leads to very good performances, which is an indication of the robustness of the search.

To complete the calibration phase, we examined if the impact of the intensification phase and the trim procedure on the solution quality.

Recall that the intensification phase is based on the idea of focusing the search on promising regions, signaled by solutions that either improve the overall best solution or are close to this value, in order to avoid bypassing solutions of good quality. On the other hand, the exact evaluation of a new configuration performed by solving the associated minimum cost network flow problem, may result in arcs that are open but do not carry flow. The *trim* procedure consists in closing these arcs. This reduces the solution value, but also modifies the basic move definition and impacts the search trajectory.

We started the investigation by running five problems using all four possible combinations of including or not the two procedures. Since the results were similar on all five problems, we decided not to continue the investigation. Figure 6 illustrates the behaviour of the four versions of the tabu search procedure on two problem instances, 20,300,200,F,L and 20,300,40,F,T (identified by the number of nodes, design arcs, commodities, a letter indicating high fixed costs, and rather loose and tight capacities, respectively). The behaviour of the four tabu search variants on these instances is typical of the general method performance and thus, in order not to overload the paper, the other graphs are not shown. The results clearly indicate that including the intensification and trim procedures is beneficial for quality of the search. Consequently, all subsequent experiments have been conducted with a tabu search that includes the two procedures.

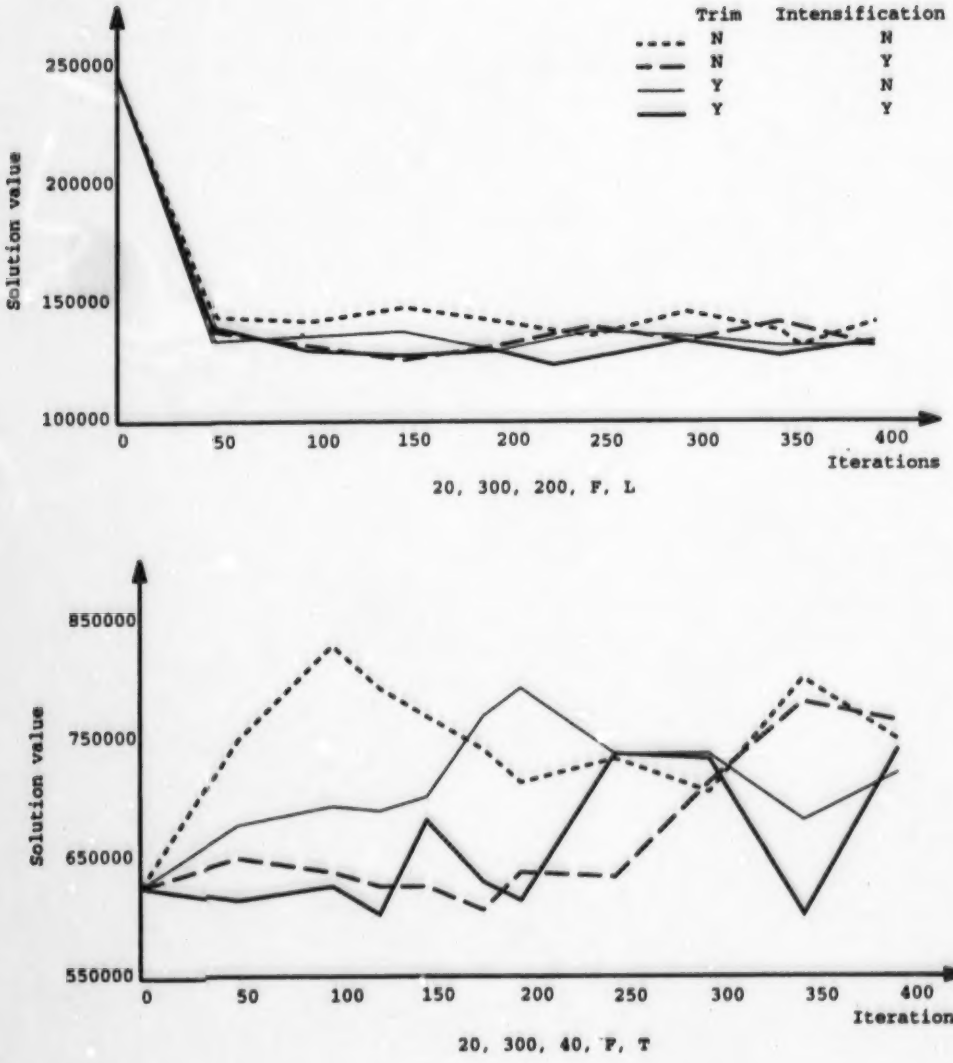


Figure 6: Calibration for Intensification and Trim

5.2 Result analysis

To evaluate the performance of the tabu search algorithm proposed in this paper, we compare its output to the optimal solution obtained using the branch-and-bound algorithm of CPLEX 6.5 (1999 REF), as well as to the results of the TABU-PATH procedure presented by Crainic, Gendreau, and Farvolden (2000), which was found to be superior to both classical greedy descent and resource decomposition (Gendron and Crainic 1996).

The same two data sets used by Crainic, Gendreau, and Farvolden (2000) are also used in this paper. Detailed results for the first set, denoted **C**, are presented in this section. Results for the second set, denoted **R** are detailed in the Appendix and summarized in the following. Problems in both sets are general transshipment networks with no parallel arcs, single origin-destination pair for each commodity, and unique but arc-specific commodity routing costs. Problem instances have been generated to offer for each network size a variety of fixed to routing cost and capacity to demand ratios. A detailed description of problem instances is given in Gendron and Crainic (1994, 1996). The problem generators as well as the problem instances can be obtained from the authors.

Tables 2 and 3 display results for the first set, denoted **C**. The problems are identified in the first column by the number of nodes, arcs, and commodities,as well as two letters summarizing the fixed cost and capacity information: a relatively high or low fixed cost relative to the routing cost is signaled by the letter **F** or **L**, respectively, while letters **T** and **L** indicate respectively if the problem is tightly or somewhat loosely capacitated compared to the total demand. The OPT column corresponds to the solution of the the branch-and-bound algorithm solved using CPLEX 6.5 (1999) on one 400MHz processor of a 64-CPU Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7. A limit of 10 hours was imposed. An **X** indicates that the procedure has failed to produce a feasible solution within this time limit, while a **t** indicates that the procedure stopped due to a time limit condition.

Column TABU-PATH holds the results found by the path-based tabu search of Crainic, Gendreau, and Farvolden (2000) on the same workstations. For a valid comparison between the two approaches, two results are displayed for the cycle-based tabu search meta-heuristic. The TABU-CYCLE column displays solutions found by our approach using the same CPU time reported for the path-based method. To illustrate and analyze the behaviour of the algorithm we propose when longer runs are performed, column TC(400) displays computational results after 400 iterations. For all methods, the figures in parentheses represent total computational time in CPU seconds on the appropriate computers. For the three meta-heuristic results, the next column displays the optimality GAP relative to the branch-and-bound solution. When branch-and-bound has failed to identify a feasible solution, as in the last part of Table 3, the gap relative to the path-based tabu search is displayed instead.

A first conclusion that emerges from the two tables is that fixed cost, capacitated, multicommodity network design problems are indeed difficult to solve, as indicated by the per-

Table 2: Computational Results, C problems

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE	GAP	TC(400)	GAP
25,100,10,V,L	14712	14712	0%	14712	0%	14712	0%
	(0.36)	(5.60)		(5.60)		(48.88)	
25,100,10,F,L	14941	15889	6.34%	14941	0%	14941	0%
	(53.64)	(8.37)		(8.37)		(53.77)	
25,100,10,F,T	49899	51654	3.52%	50767	1.74%	49899	0%
	(40.58)	(17.10)		(17.10)		(51.21)	
25,100,30,V,T	365272	365272	0%	365385	0.03%	365385	0.03%
	(16.62)	(16.57)		(16.57)		(223.67)	
25,100,30,F,L	37055	38804	4.72%	38679	4.38%	37583	1.42%
	(1727.96)	(33.01)		(33.01)		(215.38)	
25,100,30,F,T	85530	86445	1.07%	86296	0.90%	86296	0.90%
	(534.18)	(71.84)		(71.84)		(224.64)	
20,230,40,V,L	423848	425046	0.28%	425091	0.29%	424778	0.22%
	(10.43)	(71.29)		(71.29)		(370.33)	
20,230,40,V,T	371475	371816	0.09%	372583	0.30%	371893	0.11%
	(52.37)	(90.28)		(90.28)		(435.61)	
20,230,40,F,T	643036	644172	0.18%	646786	0.58%	645812	0.43%
	(671.76)	(121.79)		(121.79)		(423.32)	
20,300,40,V,L	429398	429912	0.12%	429837	0.10%	429535	0.03%
	(3.76)	(71.05)		(71.05)		(611.5)	
20,300,40,F,L	586077	589190	0.53%	593323	1.24%	593322	1.24%
	(145.55)	(113.44)		(113.44)		(581.94)	
20,300,40,V,T	464509	464509	0%	466525	0.43%	464724	0.05%
	(83.88)	(145.33)		(145.33)		(589.64)	
20,300,40,F,T	604198	606364	0.36%	609285	0.84%	607100	0.48%
	(59.88)	(123.42)		(123.42)		(560.38)	
20,230,200,V,L	94386	122592	29.88%	103930	10.11%	98995	4.88%
	(t)	(504.50)		(504.50)		(2663.24)	
20,230,200,F,L	141737.4	188590	33.06%	154404	8.94%	146535	3.38%
	(t)	(491.63)		(491.63)		(2718.31)	
20,230,200,V,T	97914	118057	20.57%	108274	10.58%	104752	6.98%
	(t)	(548.36)		(548.36)		(2565.71)	
20,230,200,F,T	137271	182829	33.19%	153455	11.79%	147385	7.37%
	(t)	(889.69)		(889.69)		(3120.14)	
20,300,200,V,L	74972.4	88398	17.91%	81628	8.88%	80819	7.80%
	(t)	(982.21)		(982.21)		(4086.83)	
20,300,200,F,L	117306	151317	28.99%	129600	10.48%	123347	5.15%
	(t)	(1316.75)		(1316.75)		(4367.88)	
20,300,200,V,T	74991	82724	10.31%	80510	7.36%	79619	6.17%
	(t)	(938.29)		(938.29)		(3807.93)	
20,300,200,F,T	108252	135593	25.26%	122547	13.21%	114484	5.76%
	(t)	(1065.88)		(1065.88)		(4657.54)	

formance of a state-of-the-art mixed integer programming solver. Experimental results also indicate clearly that TABU-CYCLE outperforms TABU-PATH for an equivalent computational effort in almost all experiments. Thus, out of 43 problem instances, only for 9, relatively small, problems are the solutions of TABU-PATH better than those of TABU-CYCLE. Moreover, the superiority of TABU-CYCLE appears to be greater when fixed cost are high: the optimality gap of TABU-CYCLE for these problems is at most 14%, while it reaches some 33% for TABU-PATH. The average percentage improvement of TABU-CYCLE compared to TABU-PATH is around 3.36%, with a maximum improvement of 18.13%. These results are a first indication of the effectiveness of the cycle-based neighbourhood structures to generate good solutions for the CMND.

The results displayed in Tables 2 and 3 also support the hypothesis that our algorithm may identify higher quality solution given longer search times. Using 400 iterations, improved solutions were found for 35 out of the 41 problems (2 solutions were already optimal). Moreover, the benefits obtained from increasing the computational effort appear to become greater for larger instances. Thus, for example, for problems with 200 commodities the maximum optimality gap is decreased from 13.21% to 7.80%. Compared to TABU-PATH, the average improvement obtained by TC(400) is increased to 4.93% and the maximum improvement is increased to 22.30%. The results discussed in the rest of the section are obtained by using 400 iterations of the cycle-based tabu search.

A more in-depth study of the performance of the method we propose was conducted using the second set of problem instances, denoted **R**. These problems were generated to facilitate the study of the impact of particular problem characteristics on algorithmic performance. The number of nodes, design arcs, and commodities are systematically varied, one at a time and for each of these networks, nine problem instances are created by combining three levels of fixed cost ratio and three levels of capacity ratio. The fixed cost ratios considered are F01 = 0.01, F05 = 0.05, and F10 = 0.10 corresponding to continuously higher levels of fixed costs compared to the routing costs. For capacity, the ratios considered are C1 = 1, C2 = 2, and C8 = 8, indicating that the total capacity available becomes increasingly tighter relatively to the total demand. Thus, the easiest problems generally have F01 and C1 ratios, while the most difficult ones have F10 and C8 characteristics. Detailed results are included in the Appendix. Aggregated statistics are used in the following to support the discussion.

Table 4 displays the distribution of the optimality gap relative to CPLEX branch-and-bound for the two sets of problem instances, obtained by the cycle-based tabu search after 400 iterations. The first column identifies the problems set, the second column indicates the number of problems where branch-and-bound did not find a feasible solution (in 10 hours), the third indicates the number of optimal solutions found by the meta-heuristic, while the fourth displays the number of problems where the heuristic solution is better than the one found by branch-and-bound after 10 hours of computation. The next six columns correspond to the indicated gap intervals. For problems in set **C**, the optimality gap never exceed 8% and is often much smaller than 2%, for an average within 2.10% of the best solutions found by branch-and-bound. For the 153 problems of set **R**, 28 optimum solutions were found, 110 solutions are within 8%, 79 of which are within 4%, while only 7 solutions display a

Table 3: Computational Results, C problems

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE	GAP	TC(400)	GAP
100,400,10,V,L	28423	28485	0.22%	28708	1.00%	28677	0.89%
	(84.81)	(32.66)		(32.66)		(336.34)	
100,400,10,F,L	24436	24912	1.95%	24576	0.57%	23949	- 1.99%
	(t)	(33.00)		(33.00)		(306.79)	
100,400,10,F,T	66364	71128	7.18%	68004	2.47%	67014	0.98%
	(t)	(81.23)		(81.23)		(626.46)	
100,400,30,V,T	385544	385185	- 0.09%	385508	- 0.01%	385508	- 0.01%
	(t)	(277.50)		(277.50)		(1975.34)	
100,400,30,F,L	50496	58773	16.39%	55401	9.71%	51552	2.09%
	(t)	(100.16)		(100.16)		(1300.56)	
100,400,30,F,T	141278	149282	5.67%	146455	3.66%	145144	2.74%
	(t)	(215.71)		(215.71)		(1869.98)	
30,520,100,V,L	53966	56426	4.56%	55358	2.58%	54958	1.84%
	(t)	(995.64)		(995.64)		(3355.96)	
30,520,100,F,L	95294	104117	9.26%	103747	8.87%	99586	4.50%
	(t)	(939.24)		(939.24)		(4032.35)	
30,520,100,V,T	52085	53288	2.31%	52985	1.73%	52985	1.73%
	(t)	(1218.52)		(1218.52)		(3481.08)	
30,520,100,F,T	98357	107894	9.70%	106877	8.66%	105523	7.29%
	(t)	(670.29)		(670.29)		(3927.35)	
30,700,100,V,L	47603	48984	2.90%	48824	2.56%	48398	1.67%
	(1736.05)	(1265.11)		(1265.11)		(4396.42)	
30,700,100,F,L	60525	65356	7.98%	63302	4.59%	62471	3.22%
	(t)	(1479.59)		(1479.59)		(4755.01)	
30,700,100,V,T	45944.5	47083	2.48%	47025	2.35%	47025	2.35%
	(t)	(2426.02)		(2426.02)		(4560.11)	
30,700,100,F,T	55709	58804	5.56%	57886	3.91%	57886	3.91%
	(t)	(1735.72)		(1735.72)		(4866.11)	
30,520,400,V,L	112997.5	125831	11.36%	122048	8.01%	120652	6.77%
	(t)	(5789.27)		(5789.27)		(36530.8)	
30,520,400,F,L	X	177409	-	167837	- 5.40%	161098	- 9.19%
	(t)	(6406.62)		(6406.62)		(429296)	
30,520,400,V,T	X	125518	-	121603	- 3.12%	121588	- 3.13%
	(t)	(6522.23)		(6522.23)		(28214)	
30,520,400,F,T	X	174526	-	171641	- 1.65%	167939	- 3.77%
	(t)	(8415.24)		(8415.24)		(40010.9)	
30,700,400,V,L	X	110000	-	108029	- 1.79%	106777	- 2.93%
	(t)	(12636.2)		(12636.2)		(24816.8)	
30,700,400,F,L	X	165484	-	154215	- 6.81%	148950	- 9.99%
	(t)	(11367.70)		(11367.70)		(69540.1)	
30,700,400,V,T	X	103768	-	102468	- 1.25%	101672	- 2.02%
	(t)	(15879.50)		(15879.50)		(34974.9)	
30,700,400,F,T	X	150919	-	148243	- 1.77%	142778	- 5.39%
	(t)	(11660.40)		(11660.40)		(51877.9)	

gap greater than 10%. The average gap over all **R** problems is 2.97%. This constitutes a very good performance, especially given the dimensions and difficulty of the most problem instances treated.

Table 4: Distribution of Relative Gaps

P.Set	X	OPT	IMP	(0-2%]	(2-4%]	(4-6%]	(6-8%]	(8-10%]	>10%
C	7	3	2	15	6	4	6	-	-
R	-	28	-	46	33	20	11	8	7

Table 5 and 6 summarize information on the behaviour of the cycle-based tabu search according to various problem characteristics. This information appears under the heading TABU-CYCLE(400). These results are also contrasted to the corresponding behaviour of the path-based tabu search. The statistics are computed over the problem instances of set **R**.

Table 5 displays the optimality gap distributions of TABU-PATH and TABU-CYCLE(400) according to the fixed cost and the capacity ratios. One notices, as expected, more tightly constrained problems are more difficult to solve and thus, for the same computational effort, somewhat less good results are obtained. Compared to the path-based method, however, the cycle-based procedure displays a much more consistent behaviour and appears significantly more robust with respect to variation in these two characteristics. Thus, for example, the average gap is not larger than 4.40% contrasted to the almost 20% of TABU-PATH. In fact, the cycle-based methods cuts the optimality gaps in all cases, significantly more so (by at least a factor of 2) as problems become more difficult.

Table 5: Gap Distribution According to Fixed Cost and Capacity Level

TABU-PATH			
	C1	C2	C8
F01	2.49%	2.21%	2.96%
F05	12.30%	9.16%	6.33%
F10	19.86%	14.45%	8.6%

TABU-CYCLE(400)			
	C1	C2	C8
F01	1.48%	1.31%	1.74%
F05	3.49%	3.69%	4.14%
F10	3.55%	4.27%	4.40%

Table 6 displays results for the distribution of optimality gaps according to problem dimensions. The same general behaviour as discussed previously is also observed in this case. For the same computation effort, larger problem instances are, as expected, more difficult to address. On the other hand, this difficulty increases slowly, much slowly than for the path-based procedure. In fact, one observes again the robustness of the cycle-based meta-heuristic with respect to problem dimensions, including the number of commodities. All these results confirm the interest of cycle-based neighbourhoods and the superiority of TABU-CYCLE in addressing CMND problems.

Summarizing the computational results, after testing on 196 problem instances, we conclude that cycle-based neighbourhoods help build meta-heuristics that yield very good solutions to the CMND. The TABU-CYCLE improves almost all TABU-PATH solutions. The

Table 6: Gap Distribution According to Problem Dimensions

TABU-PATH						
$\|P\|$	$\|N\|, \|A\|$	opt Gap	$\|N\|, \|A\|$	Opt Gap	$\|N\|, \|A\|$	Opt Gap
10		0.61%		1.81%		2.04%
25	10,25	0.56%	10,50	2.78%	10,75	4.03%
50		1.54%		7.69%		8.96%
TABU-CYCLE(400)						
10		0.00%		0.10%		0.41%
25	10,25	0.78%	10,50	0.48%	10,75	0.91%
50		1.63%		2.47%		2.87%
TABU-PATH						
$\|P\|$	$\|N\|, \|A\|$	Opt Gap	$\|N\|, \|A\|$	Opt Gap	$\|N\|, \|A\|$	Opt Gap
40		5.00%		8.43%		6.47%
100	20,100	9.03%	20,200	16.87%	20,300	20.8%
200		8.06%		24.4%		23.24%
TABU-CYCLE(400)						
40		2.78%		3.50%		2.90%
100	20,100	2.69%	20,200	6.32%	20,300	5.47%
200		5.12%		6.82%		8.19%

average improvement is around 4.75% while the maximum improvement is 33.7%. Compared to state-of-the-art branch-and-bound methods, the method finds very good quality solutions with a reasonable computation effort. More importantly, cycle-based tabu search displays a robust behaviour relative to variations in problem dimension, including number of commodities, fixed to routing cost ratios, and capacity tightness levels.

6 Conclusions

In this paper, we proposed new cycle-based neighbourhood structures for the fixed-charge capacitated multicommodity network design problem. These neighbourhoods are different from traditional ones (add/drop/swap) used in the literature for combinatorial optimization problems. They allow to identify moves based on potential variations in the the flow distribution of several commodities and that impact the status of several design arcs simultaneously.

To evaluate the quality of these neighbourhoods, a simple tabu search method has been developed. Numerical experiments have shown this method to display a consistently superior and robust performance. It generates solutions of high quality and outperforms other heuristics proposed in the literature. It is, in fact, the best current heuristic for fixed charge, capacitated, multicommodity network design problems.

Several interesting research avenues are now before us. One avenue consists in studying the impact of the linear programming methodology used to solve the minimum cost network flow problem on the efficiency and quality of the method. This may be particularly important as increasingly larger problem instances are addressed. Parallelization strategies also contribute toward this goal. A second development direction consists in building a more complete tabu search meta-heuristic that uses cycle-based neighbourhoods. Long-term memory-based strategies (e.g., path relinking and scatter sesarch) will be investigated as efficient means to diversify the sesarch and improve its performance. Finally, the application of cycle-based neighbourhoods to other difficult combinatorial optimization problems opens up intriguing but challenging perspectives. We have undertaken efforts in several of these directions and expect to report on them soon.

Acknowledgments

Funding for this project has been provided by the Natural Sciences and Engineering Council of Canada through its Research Grant program and through the Network for Mathematical Modelling and Computation, and by the Fonds F.C.A.R. of the Province of Québec.

References

Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993). *Network Flows - Theory, Algorithms, and Applications*. Prentice-Hall, Englewood Cliffs, NJ.

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

cplex (1999). *ILOG CPLEX 6.5*. ILOG, Mountain View, CA. U.S.A.

Crainic, T.G., Frangioni, A., and Gendron, B. (2000). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*. forthcoming.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (1998). A Simplex-Based Tabu Search Method for Capacitated Network Design. Publication CRT-98-37, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3):223–236.

Crainic, T.G., Gendreau, M., Soriano, P., and Toulouse, M. (1993). A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements. *Annals of Operations Research*, 41:359–383.

Crainic, T.G. and Rousseau, J.-M. (1986). Multicommodity, Multimode Freight Transportation: A General Modeling and Algorithmic Framework for the Service Network Design Problem. *Transportation Research B: Methodology*, 20B:225–242.

Farvolden, J.M. and Powell, W.B. (1994). Subgradient Methods for the Service Network Design Problem. *Transportation Science*, 28(3):256–272.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B. and Crainic, T.G. (1996). Bounding Procedures for Multicommodity Capacitated Network Design Problems. Publication CRT-96-06, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer, Norwell, MA.

Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers & Operations Research*, 1(3):533–549.

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing*, 1(3):190–206.

Glover, F. (1990). Tabu Search – Part II. *ORSA Journal on Computing*, 2(1):4–32.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Kluwer, Norwell, MA.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Desing Problem. *Operations Research*, 48(3):461–481.

Jarvis, J.J. and Mejia de Martinez, O. (1977). A Sensitivity Analysis of Multicommodity Network Flows. *Transportation Science*, 11(4):299–306.

Koskosidis, Y.A., Powell, W.B., and Solomon, M.M. (1992). An Optimization–based Heuristic for Vehicle Routing and Scheduling with Soft Time Windows Constraints. *Transportation Science*, 26:69–85.

Magnanti, T.L. and Wong, R.T. (1986). Network Design and Transportation Planning: Models and Algorithms. *Transportation Science*, 18(1):1–55.

Minoux, M. (1986). Network Synthesis and Optimum Network Design Problems: Models, Solution Methods and Applications. *Networks*, 19:313–360.

Powell, W.B. (1986). A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science*, 20(4):246–357.

Appendix

Results for each problem instance of set **R** are given in Tables 7 to 24. The following information is displayed:

- OPT: The optimal solution obtained by using the branch-and-bound algorithm of CPLEX version 6.5, on one 400MHz processor of a 64-processors Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 2.7, and limited to 10 hours CPU time. In this column, a **t** indicates that the procedure stopped due to the time limit condition.
- TABU-PATH: The value of the solution obtained by the path-based tabu search meta-heuristic on SUN Ultra Sparc1/140 workstations with 64MB of RAM memory, as reported by Crainic, Gendreau, and Farvolden (1998).
- TABU-CYCLE(400): The value of the solution obtained by the cycle-based tabu search meta-heuristic when performing 400 iterations.
- GAP: The optimality gaps of TABU-PATH and TABU-CYCLE(400) relative to branch-and-bound solutions.
- The figures in parentheses represent total computation time in CPU seconds on the appropriate machines.

Table 7: Computational Results R01: 10,25,10

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R01,F01,C1	74079	74079	0.00%	74079	0.00%
	(0.04)	(5.08)		(8.68)	
R01,F05,C1	92403	92403	0.00%	92403	0.00%
	(0.04)	(5.18)		(8.97)	
R01,F10,C1	115304	115304	0.00%	115304	0.00%
	(0.09)	(4.59)		(8.88)	
R01,F01,C2	84908	84908	0.00%	84908	0.00%
	(0.46)	(8.80)		(9.62)	
R01,F05,C2	113036	114565	1.35%	113036	0.00%
	(1.35)	(8.24)		(11.84)	
R01,F10,C2	147599	151078	2.36%	147599	0.00%
	(2.36)	(8.88)		(11.49)	

Table 8: Computational Results R02: 10,25,25

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R02,F01,C1	232239	232239	0.00%	232239	0.00%
	(0.31)	(18.54)		(19.31)	
R02,F05,C1	322453	326333	1.20%	328005	1.72%
	(1.12)	(16.43)		(25.99)	
R02,F10,C1	419503	424512	1.19%	426866	1.76%
	(1.37)	(11.47)		(26.07)	
R02,F01,C2	316437	316437	0.00%	316437	0.00%
	(0.23)	(29.23)		(27.55)	
R02,F05,C2	431250	431889	0.15%	433442	0.51%
	(0.53)	(29.23)		(34.9)	
R02,F10,C2	559578	564349	0.85%	563570	0.71%
	(0.55)	(26.88)		(29.85)	

Table 9: Computational Results R03:10,25,50

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R03,F01,C1	484830	484830	0.00%	484830	0.00%
	(0.67)	(44.38)		(47.73)	
R03,F05,C1	703362	710840	1.06%	712008	1.23%
	(2.01)	(42.10)		(55.19)	
R03,F10,C1	944990	965330	2.15%	981656	3.88%
	(2.34)	(26.83)		(59.27)	
R03,F01,C2	704247	704247	0.00%	706223	0.28%
	(0.47)	(44.30)		(59.57)	
R03,F05,C2	932897	932897	0.00%	953877	2.25%
	(1.26)	(48.69)		(59.56)	
R03,F10,C2	1188638	1188796	0.01%	1214120	2.14%
	(0.98)	(47.70)		(73.47)	

Table 10: Computational Results R04:10,50,10

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R04,F01,C1	31730	31730	0.00%	31730	0.00%
	(0.02)	(4.10)		(11.51)	
R04,F05,C1	48920	50887	4.02%	48920	0.00%
	(0.06)	(5.39)		(13.36	
R04,F10,C1	63767	67128	5.27%	63767	0.00%
	(0.06)	(5.16)		(12.34)	
R04,F01,C2	33740	33740	0.00%	33740	0.00%
	(0.57)	(8.03)		(13.85)	
R04,F05,C2	53790	53911	0.22%	53790	0.00%
	(2.62)	(6.19)		(14.45)	
R04,F10,C2	74030	75109	1.46%	74030	0.00%
	(1.72)	(6.05)		(14.01)	
R04,F01,C8	68291.7	68310	0.03%	68293	0.00%
	(5.14)	(28.49)		(18.86)	
R04,F05,C8	113004	113283	0.25%	113226	0.20%
	(9.41)	(20.10)		(18.29)	
R04,F10,C8	163208	171481	5.07%	164430	0.75%
	(7.61)	(14.43)		(18.32)	

Table 11: Computational Results R05:10,50,25

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R05,F01,C1	123003	123061	0.05%	123003	0.00%
	(0.38)	(11.85)		(30.41)	
R05,F05,C1	170060	173785	2.19%	170467	0.24%
	(1.51)	(12.84)		(29.98)	
R05,F10,C1	221486	231238	4.40%	221486	0.00%
	(8.44)	(13.47)		(29.4)	
R05,F01,C2	131608	131772	0.12%	131608	0.00%
	(1.58)	(17.37)		(33.54)	
R05,F05,C2	204157	213748	4.70%	205764	0.79%
	(15.60)	(23.23)		(35.08)	
R05,F10,C2	286524	314326	9.70%	292244	2.00%
	(65.12)	(25.91)		(36.49)	
R05,F01,C8	278372	279251	0.32%	278372	0.00%
	(1.45)	(61.25)		(63.25)	
R05,F05,C8	445810	456144	2.32%	449477	0.82%
	(1.99)	(53.78)		(66.36)	
R05,F10,C8	625879	633282	1.18%	629040	0.51%
	(2.20)	(62.49)		(69.07)	

Table 12: Computational Results R06:10,50,50

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R06,F01,C1	245936	248394	1.00%	248615	1.09%
	(3.97)	(47.12)		(56.43)	
R06,F05,C1	401685	426820	6.26%	412283	2.64%
	(51.12)	(47.12)		(66.28)	
R06,F10,C1	559477	676693	20.95%	578752	3.45%
	(205.18)	(35.61)		(66.24)	
R06,F01,C2	286682	289572	1.01%	288460	0.62%
	(25.18)	(78.56)		(69.59)	
R06,F05,C2	498266	534210	7.21%	515967	3.55%
	(463.4)	(47.05)		(80.83)	
R06,F10,C2	734414	846009	15.20%	771683	5.07%
	(1855.94)	(44.00)		(79.68)	
R06,F01,C8	682921	682921	0.00%	683614	0.10%
	(1.47)	(119.29)		(155.22)	
R06,F05,C8	1030479	1042968	1.21%	1051780	2.07%
	(1.45)	(131.228)		(153.05)	
R06,F10,C8	423316	492588	16.36%	438860	3.67%
	(46.56)	(24.5415)		(55.54)	

Table 13: Computational Results R07:10,75,10

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R07,F01,C1	32807	32807	0.00%	32807	0.00%
	(0.08)	(6.24)		(15.65)	
R07,F05,C1	47252	47252	0.00%	47252	0.00%
	(0.13)	(7.30)		(15.3)	
R07,F10,C1	62962	62962	0.00%	62962	0.00%
	(0.62)	(9.31)		(16.25)	
R07,F01,C2	37432	37432	0.00%	37432	0.00%
	(0.66)	(11.35)		(17.04)	
R07,F05,C2	56475	56916	0.78%	56591	0.21%
	(4.39)	(11.20)		(16.85)	
R07,F10,C2	77249	79473	2.88%	78875	2.10%
	(19.01)	(12.17)		(16.99)	
R07,F01,C8	59947	61896	3.25%	59947	0.00%
	(11.05)	(23.01)		(21.84)	
R07,F05,C8	99194	104454	5.30%	100155	0.97%
	(21.75)	(18.94)		(21.57)	
R07,F10,C8	141692	150413	6.15%	142319	0.44%
	(54.39)	(20.40)		(20.61)	

Table 14: Computational Results R08: 10,75,25

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R08,F01,C1	102531	102694	0.16%	102556	0.02%
	(1.33)	(24.87)		(39.13)	
R08,F05,C1	143894	148298	3.06%	143894	0.00%
	(0.54)	(18.64)		(36.77)	
R08,F10,C1	182793	192206	5.15%	182793	0.00%
	(1.35)	(19.77)		(43.04)	
R08,F01,C2	109325	110759	1.31%	109325	0.00%
	(5.45)	(29.33)		(38.43)	
R08,F05,C2	157047	161148	2.61%	158168	0.71%
	(34.23)	(20.56)		(49.01)	
R08,F10,C2	207540	218926	5.49%	208135	0.29%
	(34.23)	(23.06)		(49.54)	
R08,F01,C8	154160	156854	1.75%	155384	0.79%
	(75.38)	(68.92)		(64.34)	
R08,F05,C8	274866.5	296412	7.84%	283133	3.01%
	(404.49)	(39.64)		(49.17)	
R08,F10,C8	415793	452904	8.93%	429896	3.39%
	(776.56)	(54.99)		(74.05)	

Table 15: Computational Results R09: 10,75,50

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R09,F01,C1	171512	172484	0.57%	172343	0.48%
	(2.33)	(37.79)		(73.11)	
R09,F05,C1	296712	313486	5.65%	307038	3.48%
	(32.65)	(32.67)		(76.14)	
R09,F10,C1	424266	546647	28.85%	435590	2.67%
	(218.77)	(30.05)		(84.05)	
R09,F01,C2	192736	195129	1.24%	193242	0.26%
	(10.2)	(77.60)		(89.68)	
R09,F05,C2	357318	400172	11.99%	371998	4.11%
	(143.32)	(61.73)		(104.21)	
R09,F10,C2	522187	644762	23.47%	555945	6.46%
	(1227.52)	(70.58)		(119.60)	
R09,F01,C8	345057	349096	1.17%	348297	0.94%
	(27.83)	(138.66)		(29.67)	
R09,F05,C8	646579	662417	2.45%	669802	3.59%
	(89.5)	662417		(136.86)	
R09,F10,C8	951136	1001559	5.30%	987938	3.87%
	(310.59)	(103.64)		(165.33)	

Table 16: Computational Results R10: 20,100,40

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R10,F01,C1	200087	201350	0.63%	200613	0.26%
	(10.72)	(62.23)		(124.15)	
R10,F05,C1	346813.5	362848	4.62%	350573	1.08%
	(788.2)	(72.47)		(135.11)	
R10,F10,C1	488015	568114	16.41%	507118	3.91%
	(563.29)	(61.78)		(161.92)	
R10,F01,C2	229196	231546	1.03%	232473	1.43%
	(184.31)	(125.36)		(80.32)	
R10,F05,C2	411664	439802	6.84%	432913	5.16%
	(5123.25)	(65.69)		(196.44)	
R10,F10,C2	609104	659189	8.22%	640621	5.17%
	(27260.53)	(95.61)		(175.35)	
R10,F01,C8	486895	489385	0.51%	488737	0.38%
	(391.33)	(704.73)		(464.73)	
R10,F05,C8	951056	993363	4.45%	980010	3.04%
	(555.52)	(421.21)		(414.63)	
R10,F10,C8	1421740	1454329	2.29%	1487270	4.61%
	(608.38)	(299.67)		(332.92)	

Table 17: Computational Results R11: 20,100,100

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R11,F01,C1	714431	726155	1.64%	725416	1.54%
	(392.46)	(423.61)		(392.6)	
R11,F05,C1	1265985	1408514	11.26%	1306090	3.17%
	(t)	(272.10)		(631.13)	
R11,F10,C1	1846295	2392241	29.57%	1914040	3.67%
	(t)	(194.946)		(674.63)	
R11,F01,C2	870451	888165	2.04%	876894	0.74%
	(2960.13)	(661.47)		(580.6)	
R11,F05,C2	1623640	1846121	13.70%	1694860	4.39%
	(21739.5)	(545.29)		(1069.78)	
R11,F10,C2	2414060	2732989	13.21%	2607690	8.02%
	(t)	(429.89)		(1244.41)	
R11,F01,C8	2294912	2308694	0.60%	2295790	0.04%
	(31.51)	(1996.29)		(1126.27)	
R11,F05,C8	3507100	3585266	2.23%	3568430	1.75%
	(79.79)	(1241.06)		(1160.19)	
R11,F10,C8	4579353	4901168	7.03%	4621900	0.93%
	(48.51)	(1262.01)		(786.53)	

Table 18: Computational Results R12: 20,100,200

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R12,F01,C1	1639443	1728210	5.41%	1713670	4.53%
	(4436.74)	(1124.61)		(3012.11)	
R12,F05,C1	3403074.2	4037454	18.64%	3746250	10.08%
	(t)	(903.331)		(3274.92)	
R12,F10,C1	5276171	6415405	21.59%	6070200	15.05%
	(t)	(718.05)		(3612.71)	
R12,F01,C2	2303557	2339682	1.57%	2326230	0.98%
	(1739.79)	(2824.39)		(4700.35)	
R12,F05,C2	4669799	4950073	6.00%	4967940	6.38%
	(3145.59)	(2013.87)		(7932.05)	
R12,F10,C2	7100019	7781907	9.60%	7638050	7.58%
	(1683.84)	(1543.92)		(4728.49)	
R12,F01,C8	7635270	7666421	0.41%	7637250	0.03%
	(78.13)	(3089.45)		(4728.29)	
R12,F05,C8	10067742	1017569	1.07%	10121700	0.54%
	(61.55)	(2870.44)		(2978.28)	
R12,F10,C8	11967768	12958760	8.28%	12079300	0.93%
	(50.29)	(3387.72)		(2284.02)	

Table 19: Computational Results R13: 20,200,40

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R13,F01,C1	142947	143588	0.45%	144138	0.83%
	(6.41)	(153.62)		(277.37)	
R13,F05,C1	263800	284943	8.01%	270316	2.47%
	(547.98)	(174.07)		(274.69)	
R13,F10,C1	365836	439085	20.02%	374999	2.50%
	(116.62)	(174.07)		(259.22)	
R13,F01,C2	150977	152873	1.26%	151513	0.36%
	(90.14)	(215.69)		(252.32)	
R13,F05,C2	282682	308081	8.99%	291510	3.12%
	(3551.86)	(416.29)		(285.3)	
R13,F10,C2	406790	462407	13.67%	420028	3.25%
	(29695.59)	(165.62)		(310.77)	
R13,F01,C8	208088	214345	3.01%	212451	2.10%
	(t)	(416.29)		(393.53)	
R13,F05,C8	446997	491210	9.89%	484112	8.30%
	(t)	(309.69)		(483.27)	
R13,F10,C8	698280	772063	10.57%	758715	8.65%
	(t)	(314.57)		(504.46)	

Table 20: Computational Results R14: 20,200,100

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R14,F01,C1	403414	420837	4.32%	415119	2.90%
	(495.72)	(553.60)		(615.23)	
R14,F05,C1	750091	961760	28.22%	803356	7.10%
	(t)	(518.04)		(822.76)	
R14,F10,C1	1078595	1386426	28.54%	1155840	7.16%
	(t)	(392.21)		(765.56)	
R14,F01,C2	437607	452971	3.51%	453204	3.56%
	(1327.37)	(550.10)		(665.4)	
R14,F05,C2	852330	1003389	17.72%	912456	7.05%
	(t)	(471.52)		(971.27)	
R14,F10,C2	1230749.7	1630263	32.46%	1333440	8.34%
	(t)	(435.05)		(862.72)	
R14,F01,C8	670064	728766	8.76%	702226	4.80%
	(t)	(1688.84)		(1814.58)	
R14,F05,C8	1643376	1873897	14.03%	1748930	6.42%
	(t)	(1688.84)		(3793.69)	
R14,F10,C8	2630332	3006180	14.29%	2882710	9.59%
	(t)	(1455.10)		(3858.42)	

Table 21: Computational Results R15: 20,200,200

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R15,F01,C1	1000787	1087543	8.67%	1049360	4.85%
	(5384.72)	(1209.74)		(1767.13)	
R15,F05,C1	1979413	2912747	47.15%	2158720	9.06%
	(t)	(1025.71)		(2521.43)	
R15,F10,C1	2949264	4729421	60.36%	3135760	6.32%
	(t)	(842.37)		(2618.36)	
R15,F01,C2	1148604	1253329	9.12%	1215130	5.79%
	(t)	(1487.06)		(2794.46)	
R15,F05,C2	2488753	3348931	34.56%	2756680	10.77%
	(t)	(1179.56)		(5980.84)	
R15,F10,C2	3972667	5505598	38.59%	4384640	10.37%
	(t)	(1461.27)		(5133.23)	
R15,F01,C8	2301326.6	2454883	6.67%	2355730	2.36%
	(t)	(7518.55)		(15947.9)	
R15,F05,C8	5573412.8	5839188	4.77%	5926330	6.33%
	(16107.92)	(9170.17)		(13398.5)	
R15,F10,C8	8696932	9547358	9.78%	9180920	5.57%
	(18930.96)	(8497.8)		(12288.05)	

Table 22: Computational Results R16: 20,300,40

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R16,F01,C1	136161	139134	2.18%	136538	0.28%
	(1.05)	(215.20)		(404.89)	
R16,F05,C1	239500	261086	9.01%	247682	3.42%
	(1641.30)	(284.84)		(416.9)	
R16,F10,C1	325671	372377	14.34%	338807	4.03%
	(4345.92)	(236.08)		(418.52)	
R16,F01,C2	138532	139301	0.56%	139973	1.04%
	(27.61)	(309.10)		(376.82)	
R16,F05,C2	241801	257779	6.61%	246014	1.74%
	(851.99)	(309.10)		(430.92)	
R16,F10,C2	337762	371043	9.85%	355610	5.28%
	(8173.47)	(207.44)		(421.58)	
R16,F01,C8	169502	173354	2.27%	172268	1.63%
	(t)	(392.84)		(477.06)	
R16,F05,C8	352976	381866	8.18%	365214	3.47%
	(t)	(298.00)		(520.89)	
R16,F10,C8	541626	570012	5.24%	569874	5.22%
	(t)	(359.34)		(577.86)	

Table 23: Computational Results R17: 20,300,100

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R17,F01,C1	354138	383719	8.35%	370090	4.50%
	(332.10)	(810.79)		(909.17)	
R17,F05,C1	651025	816660	25.44%	688554	5.76%
	(t)	(914.18)		(998.58)	
R17,F10,C1	917956	1323708	44.20%	971151	5.79%
	(t)	(766.44)		(999.38)	
R17,F01,C2	370590	395645	6.76%	380850	2.77%
	(2258.82)	(1199.17)		(927.25)	
R17,F05,C2	708698	865932	22.19%	753188	6.28%
	(t)	(888.50)		(1070)	
R17,F10,C2	1029242	1370038	33.11%	1108180	7.67%
	(t)	(815.94)		(1168.94)	
R17,F01,C8	505310	534490	5.77%	524038	3.71%
	(t)	(2012.43)		(1315.7)	
R17,F05,C8	1112076	1332873	19.85%	1195140	7.47%
	(t)	(1650.51)		(2508.76)	
R17,F10,C8	1847334	2246433	21.60%	1945080	5.29%
	(t)	(1626.07)		(2813.16)	

Table 24: Computational Results R18: 20,300,200

PROB	OPT	TABU-PATH	GAP	TABU-CYCLE(400)	GAP
R18,F01,C1	828559	923006	11.40%	872888	5.35%
	(t)	(1968.72)		(2339.03)	
R18,F05,C1	1542197	2247779	45.75%	1716680	11.31%
	(t)	(2181.04)		(2889.6)	
R18,F10,C1	2292684	3542160	54.50%	2377560	3.70%
	(t)	(2284.64)		(2918.59)	
R18,F01,C2	921762	1017002	10.33%	975396	5.82%
	(t)	(3465.29)		(2677.54)	
R18,F05,C2	1866215	2225384	19.25%	2037950	9.20%
	(t)	(2435.75)		(4335.1)	
R18,F10,C2	2895982	4054910	40.02%	2966370	2.43%
	(t)	(2190.01)		(4234.01)	
R18,F01,C8	1485690.8	1633508	9.95%	1622520	9.21%
	(t)	(6599.88)		(13514.9)	
R18,F05,C8	4010736.7	4454958	11.08%	4576750	14.11%
	(t)	(7160.29)		(28883.1)	
R18,F10,C8	6663506.4	7125415	6.93%	7504310	12.62%
	(t)	(6221.07)		(32695.2)	